山东大学 软件 学院

数据结构 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202000300125 | 姓名：贾星宇 | | 班级：2020级5班 |
| 实验题目：在线等价类 | | | |
| 实验学时：4 | | 实验日期： | |
| 实验目的：  1、 掌握有序链表的基本操作：插入、删除、查找。  2、 掌握链表遍历器的使用方法。 | | | |
| 硬件环境：  lntel(R) Core(TM) i5-10210U CPU @ 1.60G Hz 2 .1 1 G Hz | | | |
| 软件环境：  CLion 2020.3.3 x64 | | | |
| 实验步骤与内容：  掌握在线等价类的使用，要求使用模拟指针实现。  1、 使用模拟指针实现本实验。  2、 输入一个1-9的正整数n，代表要创建n个元素，例如输入5，则代表创建一个1,2,3,4,5组成的元素表。  3、 再输入一个大于0正整数r，代表后面要输入r个等价关系。  4、 分行输入r个等价关系，格式如(1,2)。  5、 分行输出所有等价类，一个等价类的元素由小到大依次输出。例如等价类（1,3,5,2,4），输出时排序输出（1,2,3,4,5）。  6、 如果输出不是由小到大顺序输出，解决办法很多，可以创建一个n\*n的数组，一行存放一个等价类，将所有等价类放入数组，对数组每一行元素进行排序，输出数组所有元素。 | | | |
| 结论分析与体会：  如果（a，b）∈R，则元素a和b是等价的。  等价类是指相互等价的元素的最大集合  最大意味着不存在类以外的元素与类内部的元素等价。  离线等价类问题中已知n和R，确定所有的等价类  在线等价类中，初始时有n个元素，每个元素都属于一个独立的等价类，需要：  1.把包含a和b的等价类合成一个等价类  2，确定元素ele在哪一个类中，目的是对给定的两个元素，确定是否属于同一个类。 | | | |
|  | | | |

**本次实验主要代码：**

#include <iostream>

using namespace std;

struct node{

int ele;

node \* next;

node(int \_ele,node\* \_next){

ele = \_ele;

next = \_next;

}

};

class chain{

public:

node \* firstNode;

chain(){

firstNode=NULL;

}

void addFromTail(int num){

if (firstNode==NULL){

firstNode = new node(num,NULL);

} else{

node \* p = firstNode;

while (p->next!=NULL)p = p->next;

p->next = new node(num,NULL);

}

}

void showMes(){

if (firstNode==NULL)return;

node \* p = firstNode;

cout<<"(";

while (p->next!=NULL){

cout << p->ele<<",";

p = p->next;

}

cout << p->ele<<")"<<endl;

}

void deleteEle(int index){

if (firstNode==NULL)return;

node \* p = firstNode;

node \* pre;

if (index==1){

firstNode = p->next;

delete p;

return;

}

for (int i = 1; i < index; ++i) {

pre = p;

p = p->next;

}

pre->next = p->next;

delete p;

}

node \* getTail(){

if (firstNode==NULL)return NULL;

node \* p = firstNode;

while (p->next!=NULL)p = p->next;

return p;

}

void deleTail(){

if (firstNode==NULL)return;

node \* p = firstNode;

node \* pre;

while (p->next!=NULL){

pre = p;

p = p->next;

}

if (p==firstNode){

firstNode=NULL;

} else pre->next=NULL;

delete p;

}

};

int main() {

cout <<"Input"<<endl;

int number;

cin >> number;

int rSize;

cin >> rSize;

string arr[rSize];

for (int i = 0; i < rSize; ++i) {

cin >> arr[i];

}

chain theChain[number+1];

bool isUsed[number+1];

for (int i = 1; i <= number; ++i) {

isUsed[i] = false;

}

for (int i = 0; i < rSize; ++i) {

int one = arr[i][1]-48;

int two = arr[i][3]-48;

theChain[one].addFromTail(two);

theChain[two].addFromTail(one);

}

cout << "Output" <<endl;

chain unprocessed;

for (int i = 1; i <=number; ++i) {

if (!isUsed[i]){

cout <<"("<<i;

isUsed[i] = true;

if (theChain[i].firstNode!=NULL)

unprocessed.addFromTail(i);

while (unprocessed.firstNode!=NULL){

int tailEle = unprocessed.getTail()->ele;

unprocessed.deleTail();

while (theChain[tailEle].firstNode!=NULL){

int qele = theChain[tailEle].getTail()->ele;

theChain[tailEle].deleTail();

if (!isUsed[tailEle]){

cout <<"," <<tailEle;

isUsed[tailEle] = true;

}

if (!isUsed[qele])unprocessed.addFromTail(qele);

}

}

cout <<")"<<endl;

}

}

cout << "End0"<<endl;

return 0;

}